| APPLICATION NO. | FILING DATE | FIRST NAMED INVENTOR | ATTORNEY DOCKET NO. | CONFIRMATION NO. |
|---|---|---|---|---|
| 10/665,305 | 09/22/2003 | Thomas Goering | 11884-400201 | 5450 |

| 23838 | 7590 | 08/14/2006 |
|---|---|---|

KENYON & KENYON LLP
1500 K STREET N.W.
SUITE 700
WASHINGTON, DC  20005

| EXAMINER |
|---|
| SINGH, RACHNA |

| ART UNIT | PAPER NUMBER |
|---|---|
| 2176 | |

DATE MAILED: 08/14/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

PTO-90C (Rev. 10/03)

| | Application No. | Applicant(s) |
|---|---|---|
| **Office Action Summary** | 10/665,305 | GOERING, THOMAS |
| | Examiner | Art Unit | |
| | Rachna Singh | 2176 | |

-- *The MAILING DATE of this communication appears on the cover sheet with the correspondence address* --

**Period for Reply**

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE <u>3</u> MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.
- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

**Status**

1)☒ Responsive to communication(s) filed on <u>02 June 2006</u>.

2a)☒ This action is **FINAL.**    2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

**Disposition of Claims**

4)☒ Claim(s) <u>1-19</u> is/are pending in the application.

    4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) <u>1-19</u> is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

**Application Papers**

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on <u>22 September 2003</u> is/are: a)☒ accepted or b)☐ objected to by the Examiner.

    Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

    Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

**Priority under 35 U.S.C. § 119**

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

    a)☐ All   b)☐ Some * c)☐ None of:

      1.☐ Certified copies of the priority documents have been received.

      2.☐ Certified copies of the priority documents have been received in Application No. _____.

      3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

    * See the attached detailed Office action for a list of the certified copies not received.

**Attachment(s)**

1)☐ Notice of References Cited (PTO-892)

2)☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3)☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)
    Paper No(s)/Mail Date _____.

4)☐ Interview Summary (PTO-413)
    Paper No(s)/Mail Date. _____.

5)☐ Notice of Informal Patent Application (PTO-152)

6)☐ Other: _____.

# DETAILED ACTION

1.      This action is responsive to communications:  Application filed on 06/02/06.


2.      Claims 1-19 are pending.  Claims 1, 9, 17, and 19 are independent claims.


## *Claim Rejections - 35 USC § 103*


3.      The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all

obviousness rejections set forth in this Office action:


> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.


4.      Claims 1-19 are rejected under 35 U.S.C. 103(a) as being unpatentable over

Hitchcock et al., US 2005/0080756 A1, 04/14/05 (filed 09/29/03, Provisional Application

filed on 06/04/98).


In reference to claims 1 and 9, Hitchcock teaches a method and system for a

universal forms engine allowing data sharing between customizable admissions

applications.  See abstract and page 1, paragraph [0008]-[0012].  Compare to *"a*

*computer system for generating output modules in a form-based application*

*runtime environment"*.  Hitchcock discloses the following features:

-A form engine that permits the creation and processing of customizable electronic

forms and selective sharing of information between customized forms. A user enters

data only once, and the data is shared through an extensible database between

disparate forms. The forms engine presents a form to a user for input, receives data

from the user, provides the data to the appropriate entity. The forms engine integrates

the form and the data. User information and application information are abstracted from

the coding, that is the user information and application information is stored in a way

that allows the application information and user information to be changed without

reprogramming. This abstraction allows a set of user data to be extended without

reprogramming, allows user data to be displayed in different formats, and allows the

data to be validated. See page 1, paragraphs [0011]-[00015]. The forms engine uses

the application data file to produce the requested application in HTML format for display.

The application description file can be easily modified, for example to change labels or

to add additional fields without reprogramming the forms engine. See page 5,

paragraph [0065]. The application data file can be instructed to override default values

and can be customized. See page 6, paragraph [0080]. Compare to *"a form manager*

*component configured to receive an indication that a reusable form element has*

*been changed, determine which of the output modules from a set of output*

*modules are affected by the changed form element"*.

-Creating forms, parsing data on the forms, storing data, retrieving data, and deploying

data onto other forms. New forms are automatically populated with the previously

entered data. See page 1, paragraph [0012]. The applicant database can be extended

to include new attributes without making any changes to the forms engine program or to

the application files of institutions that chose not to include the new data. The forms

engine automatically uses the application data file to produce the requested application

in HTML format for display on the applicant's browser. The application description file

can be easily modified, for example, to change labels or to add additional fields. The

appearance of the application for each institution can be changed by changing its

application description file, without reprogramming the forms engine. The completed

application is transmitted to the institution with the data in any format that the institution

prefers. The institution can therefore upload the data directly into its applicant or student

information system database, merging the information seamlessly into their existing

workflow, thereby avoiding the additional expense and errors of re-keyboarding the

information. The forms engine thus has the capability of outputting application

information universally across platforms. See page 5, paragraph [0065]. Compare to

***"a runtime manager component configured to receive a request for an output***

***module from the set of output modules and cause regeneration of the requested***

***output module"***.

The claimed "invalidate" step is a means to indicate that the current form (output

module) is not valid because of the element change. Hitchcock does not expressly

state the output module is invalidated; however, he does teach that as the user enters

or customizes data only once, and the data is shared through an extensible database

between disparate forms. The institution can therefore upload the data directly into its

applicant or student information system database, merging the information seamlessly

into their existing workflow, thereby avoiding the additional expense and errors of re-keyboarding the information. The forms engine thus has the capability of outputting application information universally across platforms.  See page 5, paragraph [0065].

Hitchcock further teaches that when an applicant subsequently applies to a different institution, a new application, customized for the different institution is presented to the applicant.  Information that was entered into the previously submitted applications is retrieved from the database and presented to the applicant as populated fields of the new application, so that the applicant is not required to enter information more than once.  The applicant can, however, change the values in the pre-populated field if desired and the new values are saved for use in subsequent applications.  See page 4, paragraph [0056].  The entry of new values in the pre-populated field invalidates the pre-populated fields throughout the other applications because the new values are now being used.  Thus the step of "invalidating" is simply a means of "noting" or "acknowledging" that the current data is not to be used.  Thus by overriding the pre-populated data in the subsequent applications, Hitchcock's system is "noting" that the old data is no longer "valid".

It would have been obvious to a person of ordinary skill in the art at the time of the invention that Hitchcock's ability to merge information and data changes to other forms would entail invalidating other related forms containing incorrect data because the Hitchcock's system is equipped with the ability to "share" data among common application elements (i.e. address info, name) in order to cut down on redundancy and avoid the additional expense and errors of re-keyboarding information in multiple forms

having the same data. See page 5, paragraph [0065] and page 1, paragraphs [0003]-

[0006]. The entry of new values in the pre-populated field invalidates the pre-populated

fields throughout the other applications because the new values are now being used.

Thus the step of "invalidating" is simply a means of "noting" or "acknowledging" that the

current data is not to be used. Thus by overriding the pre-populated data in the

subsequent applications, Hitchcock's system is "noting" that the old data is no longer

"valid".


In reference to claims 2 and 10, Hitchcock that after the applicant completes an

application for one institution, the data is saved in a database and automatically

populates fields in subsequent application forms. See abstract.


In reference to claims 3 and 11, Hitchcock teaches the Application Data File is a

specially formatted text file that acts as an application description. It is a series of

"directives" and optional arguments which the forms engine parses to build the HTML

form and to merge in user data. The directives are interpreted by means of a look-up in

a data structure that stores the directive interpretations. See page 6, paragraph [0080].


In reference to claims 4 and 12, Hitchcock does not expressly state the output

module is invalidated by marking a flag associated with the module; however, Hitchock

further teaches validating the entire set of data to ensure the data fields are complete. If

a field is incomplete, then the field will be flagged during validation. See page 8,

paragraph [0117].


In reference to claims 5 and 13, Hitchcock teaches creating forms, parsing data

on the forms, storing data, retrieving data, and deploying data onto other forms. New

forms are automatically populated with the previously entered data. See page 1,

paragraph [0012]. The applicant database can be extended to include new attributes

without making any changes to the forms engine program or to the application files of

institutions that chose not to include the new data. The forms engine automatically uses

the application data file to produce the requested application in HTML format for display

on the applicant's browser. The application description file can be easily modified, for

example, to change labels or to add additional fields. The appearance of the application

for each institution can be changed by changing its application description file, without

reprogramming the forms engine. The completed application is transmitted to the

institution with the data in any format that the institution prefers. The institution can

therefore upload the data directly into its applicant or student information system

database, merging the information seamlessly into their existing workflow, thereby

avoiding the additional expense and errors of re-keyboarding the information. The forms

engine thus has the capability of outputting application information universally across

platforms. This step would entail identifying those forms for which the changes are to

be merged. See page 5, paragraph [0065].

In reference to claims 6 and 14, Hitchcock teaches most institutions have application date windows during which applications, whether electronic or paper, for a particular term are accepted. The forms engine verifies that the application is being submitted within the allowed window. Unlike pre-printed paper applications, however, the invention provides the schools the flexibility of easily changing the application date window, so that the time to apply can be extended if the institution wants to receive additional applications. Forms engine uses data from the appropriate application data file (FIG. 14) and previously entered user data to generate a page of a form. See page 4, paragraphs [0053]-[0054].

In reference to claims 7 and 15, Hitchcock disclose a template file gives the application developer absolute freedom to quickly update the application with no need to rewrite or add program code to the forms engine. Use of templates also dramatically reduces the number of functions needed by the engine, as well as the execution overhead. The template file can be in the form of specially tagged HTML; that is, instead of a line-by-line set of directives, the template can look like HTML with embedded special tags representing the form element/variable/value to interpolate. To process the template, the forms engine need only look for <QUESTION> . . . </QUESTION> sections and parse them. Many other pieces of logic could also be embedded into the templates.

In reference to claims 8 and 16, Hitchcock teaches the application description file can be easily modified, for example, to change labels or to add additional fields. The appearance of the application for each institution can be changed by changing its application description file, without reprogramming the forms engine. The completed application is transmitted to the institution with the data in any format that the institution prefers. The institution can therefore upload the data directly into its applicant or student information system database, merging the information seamlessly into their existing workflow, thereby avoiding the additional expense and errors of re-keyboarding the information. The forms engine thus has the capability of outputting application information universally across platforms. See page 5, paragraph [0065].

In reference to claims 17-18, Hitchcock teaches an applicant database that can be extended to include new attributes without making any changes to the forms engine program or to the application files of institutions that chose not to include the new data. The forms engine automatically uses the application data file to produce the requested application in HTML format for display on the applicant's browser. The application description file can be easily modified, for example, to change labels or to add additional fields. The appearance of the application for each institution can be changed by changing its application description file, without reprogramming the forms engine. The completed application is transmitted to the institution with the data in any format that the institution prefers. The institution can therefore upload the data directly into its applicant or student information system database, merging the information seamlessly into their

existing workflow, thereby avoiding the additional expense and errors of re-keyboarding the information. The forms engine thus has the capability of outputting application information universally across platforms. See page 5, paragraph [0065]. Hitchock teaches validating the entire set of data to ensure the data fields are complete. If a field is incomplete, then the field will be flagged during validation. See page 8, paragraph [0117]. Compare to *"responsive to a call to start a form output process based on an identified form: determining whether a previously generated output module associated with the identified form in the output module library has been marked as invalid; if so: regenerating the output module;*

Hitchcock does not expressly state the output module is marked as invalid in a library; however, he does teach that as the user enters or customizes data only once, and the data is shared through an extensible database between disparate forms. The institution can therefore upload the data directly into its applicant or student information system database, merging the information seamlessly into their existing workflow, thereby avoiding the additional expense and errors of re-keyboarding the information. The forms engine thus has the capability of outputting application information universally across platforms. See page 5, paragraph [0065]. Hitchcock further teaches that when an applicant subsequently applies to a different institution, a new application, customized for the different institution is presented to the applicant. Information that was entered into the previously submitted applications is retrieved from the database and presented to the applicant as populated fields of the new application, so that the applicant is not required to enter information more than once. The applicant can,

however, change the values in the pre-populated field if desired and the new values are

saved for use in subsequent applications. See page 4, paragraph [0056]. The entry of

new values in the pre-populated field invalidates the pre-populated fields throughout the

other applications because the new values are now being used. Thus the step of

"invalidating" is simply a means of "noting" or "acknowledging" that the current data is

not to be used. Thus by overriding the pre-populated data in the subsequent

applications, Hitchcock's system is "noting" that the old data is no longer "valid".

Hitchcock does not teach ***"storing the regenerated output module in the***

***output module library along with a marker to indicate that the output module is***

***valid"***. A library is a collection of documents (or output modules). Hitchcock teaches

storing forms in an application system database (same as library). The claimed

"invalidate" step is a means to indicate that the current form (output module) is not valid

because of the element change.

Further, by flagging an incomplete field or field that does not meet the rules

specified by an institution for a particular application, Hitchock is marking the output

module as invalid and by inputting the new values and validating it as complying with

the rules specified by the institution, he is marking it as valid. See page 8, paragraph

[0117].

It would have been obvious to a person of ordinary skill in the art at the time of

the invention that Hitchcock's ability to merge information and data changes to other

forms would entail marking and invalidating other related forms containing incorrect data

because the Hitchcock's system is equipped with the ability to "share" data among

common application elements (i.e. address info, name) in order to cut down on

redundancy and avoid the additional expense and errors of re-keyboarding information

in multiple forms having the same data. See page 5, paragraph [0065] and page 1,

paragraphs [0003]-[0006]. Further, by flagging an incomplete field or field that does not

meet the rules specified by an institution for a particular application, Hitchock is marking

the output module as invalid and by inputting the new values and validating it as

complying with the rules specified by the institution, he is marking it as valid. See page

8, paragraph [0117].

In reference to claim 19, Hitchcock teaches creating forms, parsing data on the

forms, storing data, retrieving data, and deploying data onto other forms. New forms

are automatically populated with the previously entered data. See page 1, paragraph

[0012]. The applicant database can be extended to include new attributes without

making any changes to the forms engine program or to the application files of

institutions that chose not to include the new data. The forms engine automatically uses

the application data file to produce the requested application in HTML format for display

on the applicant's browser. The application description file can be easily modified, for

example, to change labels or to add additional fields. The appearance of the application

for each institution can be changed by changing its application description file, without

reprogramming the forms engine. The completed application is transmitted to the

institution with the data in any format that the institution prefers. The institution can

therefore upload the data directly into its applicant or student information system

database, merging the information seamlessly into their existing workflow, thereby

avoiding the additional expense and errors of re-keyboarding the information. The forms

engine thus has the capability of outputting application information universally across

platforms. See page 5, paragraph [0065]. A library is a collection of documents (or

output modules). Hitchcock teaches storing forms in a application system database

(same as library). Compare to *"upon revision to a form element, identifying a form*

*element membership information which forms form a form library are associated*

*with the revised form element"*

Hitchcock does not expressly state *marking each of the identified forms in the*

*form library as invalid*; however, Hitchock teaches validating the entire set of data to

ensure the data fields are complete. If a field is incomplete, then the field will be flagged

during validation. See page 8, paragraph [0117].

Although Hitchcock does not expressly state marking the identified forms as

invalid, he does teach that as the user enters or customizes data, the data is shared

through an extensible database between disparate forms. The institution can therefore

upload the data directly into its applicant or student information system database,

merging the information seamlessly into their existing workflow, thereby avoiding the

additional expense and errors of re-keyboarding the information. See page 5, paragraph

[0065]. Hitchcock further teaches that when an applicant subsequently applies to a

different institution, a new application, customized for the different institution is

presented to the applicant. Information that was entered into the previously submitted

applications is retrieved from the database and presented to the applicant as populated

fields of the new application, so that the applicant is not required to enter information

more than once. The applicant can, however, change the values in the pre-populated

field if desired and the new values are saved for use in subsequent applications. See

page 4, paragraph [0056].

Thus the entry of new values in the pre-populated field invalidates the pre-

populated fields throughout the other applications because the new values are now

being used. Thus the step of "invalidating" is simply a means of "noting" or

"acknowledging" that the current data is not to be used. Thus by overriding the pre-

populated data in the subsequent applications, Hitchcock's system is "noting" that the

old data is no longer "valid". Further, by flagging an incomplete field or field that does

not meet the rules specified by an institution for a particular application, Hitchock is

marking the output module as invalid. See page 8, paragraph [0117].

It would have been obvious to a person of ordinary skill in the art at the time of

the invention that Hitchcock's ability to merge information and data changes to other

forms would entail invalidating and marking related forms containing incorrect data as

invalid because the Hitchcock's system is equipped with the ability to "share" data

among common application elements (i.e. address info, name) in order to cut down on

redundancy and avoid the additional expense and errors of re-keyboarding information

in multiple forms having the same data. See page 5, paragraph [0065] and page 1,

paragraphs [0003]-[0006]. Futhermore, the entry of new values in the pre-populated

field invalidates the pre-populated fields throughout the other applications because the

new values are now being used.  Thus the step of "invalidating" is simply a means of

"noting" or "acknowledging" that the current data is not to be used.  Thus by overriding

the pre-populated data in the subsequent applications, Hitchcock's system is "noting" or

marking that the old data is no longer "valid".

### Response to Arguments

5.      On pages 5-6 of Remarks, Applicant argues Hitchcock fails to teach invalidating

output modules.  Examiner noted in the previous office action that the claimed

"invalidate" step is simply a means to indicate that the current form (output module) is

not valid because of the element change.  It is further noted that the invalidating step

does not require there to be a "tangible" indication that an output module is invalid.  In

other words, the "invalidating" step is abstract.

Thus, in the previous office action Examiner's position indicated that although

Hitchcock does not expressly state the output module is invalidated, he does teach that

as the user enters or customizes data, the data is shared through an extensible

database between disparate forms.  The institution can therefore upload the data

directly into its applicant or student information system database, merging the

information seamlessly into their existing workflow, thereby avoiding the additional

expense and errors of re-keyboarding the information. See page 5, paragraph [0065].

Hitchcock further teaches that when an applicant subsequently applies to a different

institution, a new application, customized for the different institution is presented to the

applicant. Information that was entered into the previously submitted applications is

retrieved from the database and presented to the applicant as populated fields of the

new application, so that the applicant is not required to enter information more than

once. The applicant can, however, change the values in the pre-populated field if

desired and the new values are saved for use in subsequent applications. See page 4,

paragraph [0056]. It is the Examiner's view that the entry of new values in the pre-

populated field invalidates the pre-populated fields throughout the other applications

because the new values are now being used. Thus the step of "invalidating" is simply a

means of "noting" or "acknowledging" that the current data is not to be used. Thus by

overriding the pre-populated data in the subsequent applications, Hitchcock's system is

"noting" that the old data is no longer "valid".


It would have been obvious to a person of ordinary skill in the art at the time of

the invention that Hitchcock's ability to merge information and data changes to other

forms would entail invalidating other related forms containing incorrect data because the

Hitchcock's system is equipped with the ability to "share" data among common

application elements (i.e. address info, name) in order to cut down on redundancy and

avoid the additional expense and errors of re-keyboarding information in multiple forms

having the same data. See page 5, paragraph [0065] and page 1, paragraphs [0003]-

[0006].

On pages 6-7 of the Remarks, Applicant argues Hitchcock does not teach regenerating invalidated output modules. Examiner disagrees. Hitchcock teaches new forms are automatically populated with the previously entered data. See page 1, paragraph [0012]. The applicant database can be extended to include new attributes without making any changes to the forms engine program or to the application files of institutions that chose not to include the new data. The forms engine automatically uses the application data file to produce the requested application in HTML format for display on the applicant's browser. The application description file can be easily modified, for example, to change labels or to add additional fields. The appearance of the application for each institution can be changed by changing its application description file, without reprogramming the forms engine. The completed application is transmitted to the institution with the data in any format that the institution prefers. The institution can therefore upload the data directly into its applicant or student information system database, merging the information seamlessly into their existing workflow, thereby avoiding the additional expense and errors of re-keyboarding the information. The forms engine thus has the capability of outputting application information universally across platforms. See page 5, paragraph [0065].

Further, it is noted that the invalidating step does not require there to be a "tangible" indication that an output module is invalid. In other words, the "invalidating" step is abstract.

On pages 7-8 of the Remarks, Applicant argues Hitchock does not teach marking forms as invalid. It is noted that the claimed "marking" is not necessarily tangible and could be in the "abstract". Further the claimed "marker" is not necessarily tangible either as a marker is used to indicate an output module is valid but could be a marker that is not displayed to the user.

Hitchock teaches validating the entire set of data to ensure the data fields are complete. If a field is incomplete, then the field will be flagged during validation. See page 8, paragraph [0117].

In the previous office action Examiner's position indicated that although Hitchcock does not expressly state the output module is invalidated, he does teach that as the user enters or customizes data, the data is shared through an extensible database between disparate forms. The institution can therefore upload the data directly into its applicant or student information system database, merging the information seamlessly into their existing workflow, thereby avoiding the additional expense and errors of re-keyboarding the information. See page 5, paragraph [0065]. Hitchcock further teaches that when an applicant subsequently applies to a different institution, a new application, customized for the different institution is presented to the applicant. Information that was entered into the previously submitted applications is retrieved from the database and presented to the applicant as populated fields of the new application, so that the applicant is not required to enter information more than once. The applicant can, however, change the values in the pre-populated field if desired and the new values are saved for use in subsequent applications. See page 4,

paragraph [0056]. It is the Examiner's view that the entry of new values in the pre-populated field invalidates the pre-populated fields throughout the other applications because the new values are now being used. Thus the step of "invalidating" is simply a means of "noting" or "acknowledging" that the current data is not to be used. Thus by overriding the pre-populated data in the subsequent applications, Hitchcock's system is "noting" that the old data is no longer "valid". Further, by flagging an incomplete field or field that does not meet the rules specified by an institution for a particular application, Hitchock is marking the output module as invalid and by inputting the new values and validating it as complying with the rules specified by the institution, he is marking it as valid. See page 8, paragraph [0117].

Further it is noted that Examiner has not taken official notice of the features argued by the Applicant. Examiner has interpreted certain terms such as "invalidating" and "marker" to be in the abstract, as in they do not produce a tangible indication of something being invalid or marked.

In view of the comments above, the rejections are maintained.

## Conclusion

6.      **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time

policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE

MONTHS from the mailing date of this action. In the event a first reply is filed within

TWO MONTHS of the mailing date of this final action and the advisory action is not

mailed until after the end of the THREE-MONTH shortened statutory period, then the

shortened statutory period will expire on the date the advisory action is mailed, and any

extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of

the advisory action. In no event, however, will the statutory period for reply expire later

than SIX MONTHS from the mailing date of this final action.


7.      Any inquiry concerning this communication or earlier communications from the

examiner should be directed to Rachna Singh whose telephone number is 571-272-

4099. The examiner can normally be reached on M-F (8:30AM-6:00PM).

If attempts to reach the examiner by telephone are unsuccessful, the examiner's

supervisor, Heather Herndon can be reached on 571-272-4136. The fax phone number

for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the

Patent Application Information Retrieval (PAIR) system. Status information for

published applications may be obtained from either Private PAIR or Public PAIR.

Status information for unpublished applications is available through Private PAIR only.

For more information about the PAIR system, see http://pair-direct.uspto.gov. Should

you have questions on access to the Private PAIR system, contact the Electronic

Business Center (EBC) at 866-217-9197 (toll-free).


RS
08/08/06

*H Herndon*

Heather R. Herndon
Supervisory Patent Examiner
Technology Center 2100